Amendments to the Claims:

The following listing of claims will replace all prior versions, and listings, of claims in the application:

1. (Currently Amended) A method for simplifying an aspect-oriented programming element embodied on a recordable medium that is compilable into instructions for operating a data processing device, the programming element having at least one part, comprising:

simplifying the programming element by reducing the programming element to a canonical expression until all of the at least one part of the programming element reaches a first simplification stage to create at least one part of a current stage simplified programming element;

determining at least one propagator for the current stage simplified programming element, the propagator being a form that is matched against terms of an appropriate stage and posting information about projections of one or more of the terms, and being described in the programming element, the propagator incrementing the value of the projections,

associating at least one projection with the current stage simplified programming element using the at least one determined propagator, the at least one projection giving information about the role of a mathematical expression in the simplification of the programming element;

simplifying the current stage simplified programming element, based at least in part on the current stage simplified programming element and the associated projections, until all of the at least one part of the current stage simplified programming element reaches a next stage to create a next stage simplified programming element.

2. (Original) The method of claim 1, further comprising compiling each stage obtained from the programming element into at least a portion of the instructions for operating the data processing device.

3. (Original) The method of claim 1, further comprising repeating the determining, associating and current stage simplifying steps using the next stage simplified programming element as the current stage simplified programming element.

4. (Original) The method of claim 3, wherein repeating the determining, associating and current stage simplifying steps comprises repeating the determining, associating and current stage simplifying steps until the next stage simplified programming element is a final stage of the programming element.

5. (Original) The method of claim 1, wherein associating the at least one projection with the current stage simplified programming element using the at least one determined propagator comprises using the at least one determined propagator to decorate the current stage simplified programming element with the at least one projection.

6. (Original) The method of claim 1, wherein each simplified programming element has at least one significance, and simplifying the current stage simplified programming element comprises determining whether, for each of the at least one part of the current simplified programming element, that part of the current simplified programming element should be reduced so that the next stage simplified programming element properly denotes the at least one significance of that part of the current simplified programming element in the next stage simplified programming element.

7. (Currently Amended) A method for executing an aspect-oriented computation on a data processing device described as a plurality of language constructs, comprising:

determining at least one propagator for the computation, the propagator being a form that is matched against terms of an appropriate stage and posting information about

projections of one or more of the terms, and being included in the language ~~constructs;~~constructs and incrementing the value of the projections;

generating a projection on the computation using the propagator, the projection specifying a second computation and giving information about the role of a mathematical expression in simplifying the language constructs;

executing the computation until a portion of the computation that is conditional on a result of the projection is reached;

simplifying the language constructs describing the computation, by reducing the language constructs to canonical expressions, sufficiently to allow the second computation specified by the projection to be executed;

executing the second computation to obtain the result for the projection; and

continuing the execution of the computation based on the obtained result for the projection.

8. (Currently Amended) A method for converting an aspect-oriented programming element into a plurality of woven code blocks, the woven code blocks compilable into instructions for operating a data processing device, comprising:

(a) identifying at least one of at least one common variable and at least one common process in the programming element;

(b) reducing the programming element to at least one significance based on the identified at least one of at least one common variable and at least one common process, the at least one significance comprising a canonical expression;

(c) incorporating the at least one significance into a first woven code block;

(d) determining zero, one or more of the incorporated significances that are susceptible to updating in subsequent steps of the method;

(e) invoking a propagator, based upon results of the determination, usable to perform any desired updates on the determined susceptible significances of the first woven code block, the propagator being a form that is matched against terms of an appropriate stage and posting information about projections of one or more of the ~~terms,~~ and incrementing the value of the projections, the projections giving information about the role of a mathematical expression in reducing the programming element;

repeating steps (a)-(e) at least once to create a subsequent woven code block based on the immediately previously created woven code block, further comprising:

(f) communicating with the propagator of at least one previously created woven code block to determine if any significances of that at least one previously created woven code block are common to the subsequent woven code block; and

(g) updating any significances in at least one of the subsequent woven code block and at least one previously created woven code block that are common to the subsequent woven code block and that at least one previously created woven code block.